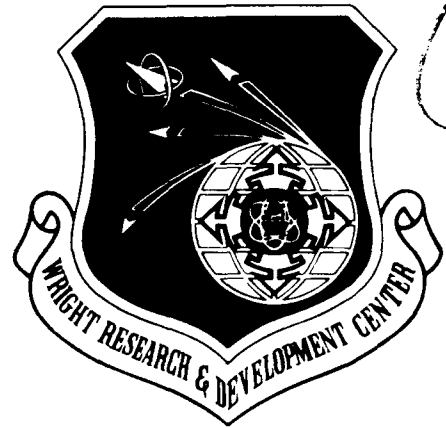


WRDC-TR-90-8007
Volume V
Part 43

AD-A250 475



INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume V - Common Data Model Subsystem
Part 43 - CDM Compare Utility Development Specification

M. Apicella, S. Singh

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209



September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533



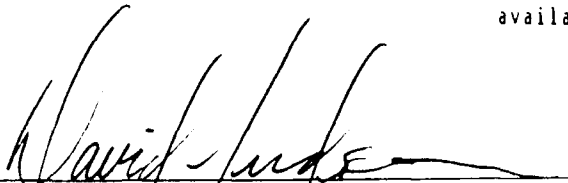
92-13411

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

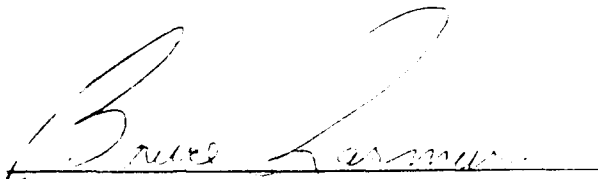
This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations


DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

FOR THE COMMANDER:


BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) DS 620341430		5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-8007 Vol. V, Part 43	
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services	6b. OFFICE SYMBOL (if applicable) WRDC/MTI	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI	
6c. ADDRESS (City, State, and ZIP Code) 2970 Presidential Drive Fairborn, OH 45324-6209		7b. ADDRESS (City, State, and ZIP Code) WPAFB, OH 45433-6533	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF	8b. OFFICE SYMBOL (if applicable) WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464	
8c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB, Ohio 45433-6533		10. SOURCE OF FUNDING NOS.	
11. TITLE (Include Security Classification) See block 19		PROGRAM ELEMENT NO. 78011F	PROJECT NO. 595600
		TASK NO. F95600	WORK UNIT NO. 20950607
12. AUTHOR(s) Control Data Corporation: Apicella, M. L., Singh, S.			
13a. TYPE OF REPORT Final Report	13b. TIME COVERED 4 / 1 / 87 - 12 / 31 / 90	14. DATE OF REPORT (Yr., Mo., Day) 1990 September 30	15. PAGE COUNT 32
16. SUPPLEMENTARY NOTES WRDC/MTI Project # 6203			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify block no.)	
FIELD	GROUP	SUB GR.	
1308	0905		
19. ABSTRACT (Continue on reverse if necessary and identify block number)			
<p>This document describes the construction of the Common Data Model (CDM) subsystem's Impact Analysis tool. This tool is used to determine what impact a software change may have on other software modules within the CDM and on the external schemas. Whatever impact occurs is reported on the terminal screen or at a printer device.</p> <p>BLOCK 11:</p> <p>INTEGRATED INFORMATION SUPPORT SYSTEM Vol V - Common Data Model Subsystem</p> <p>Part 43 - CDM Compare Utility Development Specification</p>			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson	22b. TELEPHONE NO. (Include Area Code) (513) 255-7371	22c. OFFICE SYMBOL WRDC MTI	

EDITION OF 1 JAN 73 IS OBSOLETE

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

SUBCONTRACTOR

ROLE

Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Structural Dynamics
Research Corporation

Responsible for User Interfaces,
Virtual Terminal Interface, and Network
Transaction Manager design,
development, implementation, and
support.

Arizona State University

Responsible for test bed operations
and support.

Table of Contents

		Page
SECTION	1. SCOPE	1-1
	1.1 Identification	1-1
	1.2 Functional Summary	1-1
SECTION	2. DOCUMENTS	2-1
	2.1 Reference Documents	2-1
	2.2 Terms and Abbreviations	2-1
SECTION	3. REQUIREMENTS	3-1
	3.1 Computer Program Definition	3-1
	3.1.1 System Capacities	3-1
	3.1.2 Interface Requirements	3-1
	3.1.2.1 Interface Block Diagram	3-1
	3.1.2.2 Detailed Interface Definition	3-2
	3.2 Detailed Functional Requirements	3-2
	3.2.1 CDM Extract Phase	3-2
	3.2.1.1 Inputs	3-3
	3.2.1.2 Processing	3-3
	3.2.1.3 Outputs	3-12
	3.2.2 CDM Compare Phase	3-13
	3.2.2.1 Inputs	3-14
	3.2.2.2 Processing	3-14
	3.2.2.3 Outputs	3-14
	3.2.3 CDM Compare Report Phase	3-14
	3.2.3.1 Input	3-15
	3.2.3.2 Processing	3-15
	3.2.3.3 Output	3-15
	3.3 Performance Requirements	3-16
	3.3.1 Programming Methods	3-16
	3.3.2 Modification Requirements	3-16
	3.4 Database Requirements	3-16
SECTION	4. QUALITY ASSURANCE PROVISIONS	4-1
	4.1 Introduction and Definitions	4-1
	4.2 Computer Programming Test and Evaluation	4-1
SECTION	5. PREPARATION FOR DELIVERY	5-1
APPENDIX	A. EXTRACT DATA FILE RECORD DEFINITIONS	A-1
	A.1 Conceptual Schema	A-1
	A.1.1 Category 1-Model Definition	A-1
	A.1.2 Category 2-Entity Class	A-1
	A.1.3 Category 3-Relation Class	A-1
	A.1.4 Category 4-Attribute Class	A-2
	A.1.5 Category 5-Domains	A-2
	A.2 External Schema	A-2
	A.2.1 Category 1-User Views	A-2
	A.3 Internal Schema	A-2
	A.3.1 Category 1-Database Definition	A-2
	A.4 Conceptual-External Schema Mappings	A-3
	A.4.1 Category 1-External to Conceptual Schema Mappings	A-3
	A.5 Complex Mapping	A-3
	A.5.1 Category 1-Definitions	A-3

A.6	Conceptual-Internal Schema Mappings	A-4
A.6.1	Category 1-Attribute Class Mappings	A-4
A.6.2	Category 2-Relation Class Mapping	A-4
A.6.3	Category 3-Entity Class Mappings	A-4

List of Illustrations

<u>Figure</u>	<u>Title</u>	<u>Page</u>
3-1	Extract Phase	3-1
3-2	Compare Phase	3-2
3-3	Report Phase	3-2
3-4	Extract Phase Input Screen	3-3
3-5	Record Format	3-12
3-6	Compare Phase Input Screen	3-14
3-7	Report Phase Input Screen	3-15
3-8	Report Phase Output Display	3-15

SECTION 1

SCOPE

1.1 Identification

This specification establishes the detailed requirements for performance, design, test, and qualification of a computer program identified as the CDM Compare Utility of IISS.

Please refer to the Software Availability Bulletin, Volume III, Part 16, CI# SAB620326000, for current IISS software and documentation availability.

1.2 Functional Summary

The CDM Compare Utility is used to compare two versions of a CDM and report differences in the internal, conceptual and external schemas as well as the conceptual-internal, conceptual-external and complex schema mappings.

SECTION 2

DOCUMENTS

2.1 Reference Documents

- [1] Systran, ICAM Documentation Standards, IDS150120000C, 15 September 1983.
- [2] D.Appleton Company, CDM Administrator's Manual, UM 620341000, 31 March 1988.
- [3] D.Appleton Company, CDM1, An IDEF1 Model of the Common Data Model, CCS620141000, 15 May 1985.
- [4] Control Data Corporation, Neutral Data Definition Language User's Guide, 31 March 1988.
- [5] C. J. Date, An Introduction to Database Systems, Addison-Wesley Publishing Company, Inc, 1977.
- [6] IBM, DATABASE 2 Reference release 1.0, IBM, December 1984.
- [7] Cincom Systems, TOTAL Database Administration Reference Manual, release 8.1 1978, Cincom Systems.

2.2 Terms and Abbreviations

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Common Data Model: (CDM), IISS subsystem that describes common data of an enterprise and includes conceptual, external and internal schemas and schema transformation operators.

Common Data Model Administrator: (CDMA), the person or group of persons responsible for creating and maintaining an enterprise's Common Data Model. The CDMA manages the common data rather than managing applications that access data.

Common Data Model Processor: (CDMP), a component of the Common Data Model subsystem which is the distributed database manager of the IISS.

Conceptual Schema: (CS), the standard definition used for all data in the enterprise. It is based on IDEF1 information modeling.

External Schema: (ES), an application's view of the CDM's conceptual schema.

Integrated Information Support System: (IISS), a computing environment used to investigate, demonstrate, test the concepts and produce application for information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Internal Schema: (IS), the definition of the internal model, the storage structure definition, which specifies how the physical data are stored and how they can be accessed. It is represented in terms of the physical database components, including record types and inter-record relationships.

Neutral Data Definition Language: (NDDL), A language used to manipulate and populate information in the Common Data Model (CDM) or IISS System Database.

Neutral Data Manipulation Language: (NDML), A language developed by the IISS project to provide uniform access to common data, regardless of database manager or distribution criteria. It provides distributed retrieval and single node update.

Presentation Schema: (PS), The totality of the form fields in an application which are targets of data derivative from the common data.

SECTION 3

REQUIREMENTS

3.1 Computer Program Definition

The CDM Compare Utility is used to report the differences between two versions of a CDM. Comparisons will be made on the objects within the internal, conceptual and external schemas as well as the objects in the conceptual-internal, conceptual-external schema mappings and complex mappings.

The CDM Compare Utility is composed of three parts or phases. The extract phase will obtain the pertinent information from a CDM and save the information in a file. This data extraction must be done for each version of the CDM. The compare phase will perform a comparison of the information obtained from the extract phase. The report phase displays the results of the comparison.

It is a basic assumption of this specification that both versions of the CDM will reside on the same host. Further, this implementation will be done on a VAX system. Work files identified as being dynamically created during the execution of this facility will be created with allowances for portability to other hosts.

3.1.1 System Capacities

The System capacities of the CDM Compare have not been determined.

3.1.2 Interface Requirements

The CDM Compare Utility is constrained to query database tables of the CDM and to format and present its reports via the User Interface Report Writer. NDML will be used to obtain the required information from the CDM during the extract phase. The user must be able to access the IISS environments required to access each version of the CDM.

3.1.2.1 Interface Block Diagram

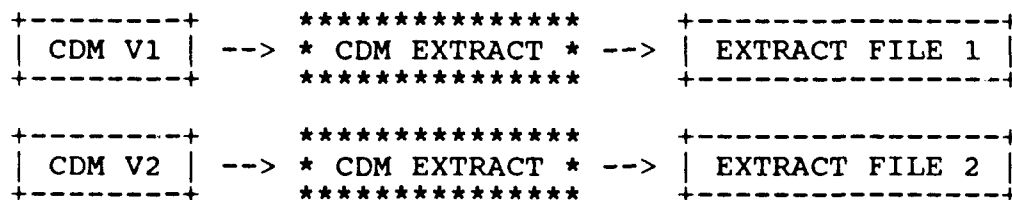


Figure 3-1 Extract Phase



Figure 3-2 Compare Phase

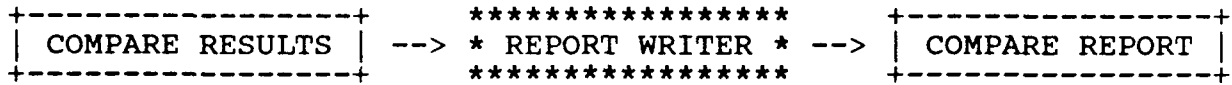


Figure 3-3 Report Phase

3.1.2.2 Detailed Interface Definition

The CDM Extract is accessed through the IISS function screen. At this step the user specifies the schemas and mappings to be compared and the name of the output data file. The information about each version of the CDM must be extracted separately as described in section 3.2. The result of a CDM Extract is a data file.

The CDM Compare is also accessed through the IISS function screen. The user specifies the names of the files containing the results of the CDM extract operation.

The CDM Compare Report is accessed through the IISS function screen. The user specifies the output medium. The report is presented via the Report Writer and may appear on any device supported by it.

3.2 Detailed Functional Requirements

The CDM Compare Utility is used to report the differences between two versions of a CDM. Comparisons will be made on the objects within the internal, conceptual and external schemas as well as the objects in the conceptual-internal, conceptual-external schema mappings and complex mappings as selected by the user.

The CDM Compare Utility is composed of three parts or phases. The extract phase will obtain the pertinent information from a CDM and save the information in a file. This data extraction must be done for each CDM. The compare phase will perform a comparison of the information obtained from the extract phase. The report phase presents the results of the comparison to a device specified by the user.

3.2.1 CDM Extract Phase

This part of the CDM Compare Utility obtains information from the two versions of the CDM, builds flattened data records and writes the records to a data file which will be used as input to the comparison phase of the CDM Compare utility. In order to

obtain the information from a given CDM, the user must enter the IISS environment which contains that CDM. This is required since an application program using NDML may access at most one version of a CDM at a time and there can only be one CDM per IISS environment.

3.2.1.1 Inputs

When the CDM Extract phase is selected from the IISS function screen, the user will be presented with the form shown in figure 3-4. The CDM Version Identifier field is a ten character field used to identify the CDM during the comparison process. The Output Data File Name is a forty character field used to name the output data file. The objects to be compared are selected by placing an "X" by the desired objects.

```
+-----+
|                                     |
|             INTEGRATED INFORMATION SUPPORT SYSTEM             |
|                                     |
|                   CDM COMPARE UTILITY                         |
|                                     |
|                   ** EXTRACT PHASE **                         |
|                                     |
| CDM Version Identifier: _____ |
|                                     |
| Output Data File Name:  _____ |
|                                     |
|           Place an "X" by each object to be compared         |
|           and press <enter> when your selections are         |
|           complete.                                           |
|                                     |
| _Internal Schema      _Conceptual-Internal Schema Mappings  |
| _Conceptual Schema    _Conceptual-External Schema Mappings  |
| _External Schema      _Complex Mapping                      |
|                                     |
| Msg: 0                                                         |
|                                     |
|                                     application                 |
+-----+
```

Figure 3-4 Extract Phase Input Screen

3.2.1.2 Processing

The CDM Extract will create a data file with the name specified by the user. The tables for those schemas and mappings selected will be read and the data written to the data file. The processing required to build the data records for each object is described below. The "Using" specification used in the processing definitions is the value in the "where" predicate of an NDML statement.

Conceptual Schema

Model definition:

Step 1. Using: N/A
Select: model_name
From: Model_Class

Entity Class, Key Definition:

Step 1. Using : N/A
Select: ec_no, model_no
From: Entity_Class

Step 2. Using: ec_no, model_no
Select: ec_name, model_name, ec_name_type,
kc_name, tag_name
From: Entity_Name, Model_Class, Key_Class
Attribute_Use_Cl

Entity Class, Owned Attributes:

Step 1. Using: N/A
Select: ec_no, model_no
From: Entity_Class

Step 2. Using: model_no, ec_no
Select: model_name, ec_name, ac_no
From: Model_Class, Entity_Class,
Owned_Attribute

Step 3. Using: ac_no
Select: ac_name
From: Attribute_Name

Entity Class, Inherited Attributes:

Step 1. Using: N/A
Select: model_no, ec_no
From: Entity_Class

Step 2. Using: model_no, ec_no
Select: model_name, ec_name, tag_no,
tag_name, ac_no
From: Model_Class, Entity_Name,
Attribute_Use_Cl

Step 3. Using: tag_no, ac_no
Select: rc_no, ac_name
From: Inherited_Att_Use, Attribute_Name

Entity Class, Independent Entity Relation Class

Step 1. Using: N/A
Select: model_no, ec_no
From: Entity_Class

Step 2. Using: model_no, ec_no
Select: model_name, ec_name, rc_name,
From: Model_Class, Entity_Name,
LINK_RELATION

Entity Class, Dependent Entity Relation Class:

Step 1. Using: N/A
Select: model_no, ec_no
From: Entity_Class

Step 2. Using: model_no, ec_no
Select: model_name, ec_name, rc_name,
From: Model_Class, Entity_Name,
LINK_RELATION

Relation Class, Relation Definition:

Step 1. Using: REL TYPE = "LINK"
Select: RC_NO
From: RELATION_CLASS

Step 2. Using: RC_NO
Select: ind_ec_no, rc_name, dep_ec_no,
no_ind_ent, min_no_dep_ent,
max_no_dep_ent
From: LINK_RELATION

Step 3. Using: ind_ec_no, dep_ec_no
Select: ec_name(ind), ec_name(dep),
model_no
From: Entity_Name, Entity_Class

Step 4. Using: ind_ec_no, model_no
Select: kc_name, kc_no, model_name
From: Key_Class, model_class

Step 5. Using: kc_no
Select: tag_name(ind), tag_name(dep)
From: Attribute_Use_Cl

Relation Class, Keyword Definition

Step 1. Using: REL TYPE = "LINK"
Select: RC_NO
From: RELATION_CLASS

Step 2. Using: RC_NO
Select: rc_name, ind_ec_no
From: LINK_RELATION

Step 3. Using: ind_ec_no, rc_no
Select: model_no, kw_no
From: Entity_Class, Rc_Keyword

Step 4. Using: model_no, kw_no
Select: model_name, Keyword
From: Model_Class, Keyword

Domain Definition - Data Type:

- Step 1. Using: N/A
Select: domain_no, domain_name
From: Domain_Class
- Step 2. Using: domain_no
Select: data_type_name, max_size,
no_of_decimals, data_type_ind
From: User_Def_Data_Type

Domain Definition, Range:

- Step 1. Using: N/A
Select: domain_no, domain_name
From: Domain_Class
- Step 2. Using: domain_no
Select: begin_value, end_value
From: Domain_Range

Domain Definition, Values:

- Step 1. Using: N/A
Select: domain_no, domain_name
From: Domain_Class
- Step 2. Using: domain_no
Select: specific_value
From: Domain_Value

External Schema

User Views, View Definitions

- Step 1. Using: N/A
Select: view_no, sec_id
From: Sec
- Step 2. Using: view_no
Select: di_id, data_type_name
From: Data_Item

INTERNAL SCHEMA

Data Base Definition:

- Step 1. Using: N/A
Select: db_id, db_name, host_id, dbms_name,
null_value
From: Data_Base
- Step 2. Using: db_id
Select: schema_name, subschema_name,
db_location
From: Schema_Names

Data Base Definition, Area Assignments

- Step 1. Using: N/A
 Select: db_id, db_name
 From: Data_Base
- Step 2. Using: db_id
 Select: area_id, rt_id
 From: Data_Base_Area, Db_Area_Assignment

Data Base Definition, Record Type Area Assignments

- Step 1. Using: N/A
 Select: db_id, db_name
 From: Data_Base
- Step 2. using: db_id
 select: rt_id
 from: Record_Type
- Step 3. Using: rt_id
 Select: area_id
 From: Db_Area_Assignment

Data Base Definition, Record Type Data Field Definition

- Step 1. Using: N/A
 Select: db_id, db_name
 From: Data_base
- Step 2. Using: db_id
 Select: rt_id
 From: Record_Type
- Step 3. Using: rt_id
 Select: df_id, rec_seq_no, rec_key_code,
 occurs, dbms_access,
 index_indicator
 From: Data_Field
- Step 4. Using: df_no
 Select: data_type, redef_df_no, component_of_df,
 occ_depend_df_no, index_by_df_no,
 filler_size
 From: Elementary_Data_Field,
 Data_Field_Redefinition,
 Component_Data_Field,
 Repeating_Df_Occ_Counter, Index_Df,
 Data_Field_Filler
- Step 5. Using: redef_field_no
 Select: df_id
 From: Data_Field
- Step 6. Using: component_of_df
 Select: df_id
 From: Data_Field

Step 7. Using: occ_depend_df_no
Select: df_id
From: Data_Field

Step 8. Using: index_by_df_no
Select: df_id
From: Data_Field

Data Base Definition, Record Type Record Set Definition:

Step 1. Using: N/A
Select: db_id, db_name
From: Data_Base

Step 2. Using: db_id
Select: set_id, rt_id_of_owner, total_num_mem
From: Record_Set

Step 3. Using: set_id
Select: rt_id_of_member, req_mem_id
From: Set_Type_Member

Step 4. Using: rt_id_of_member
Select: df_id, linkage_type
From: Df_Set_Linkage

Data Base Definition, PSB Definition:

Step 1. Using: N/A
Select: db_id, db_name
From: Data_Base

Step 2. Using: db_id
Select: psb_name, pcb_seq_no, key_feedback_len
From: Psb_Pcb

Step 3. Using: psb_name
Select: host_id
From: Psb

Conceptual - External Schema Mappings

Project Data Item Mapping

Step 1. Using: N/A
Select: view_no, view_id
From: User_View

Step 2. Using: view_no
Select: di_id
From: Data_Item

Step 3. Using: di_id
Select: tag_no, prim_secondary
From: Project_data_item

Step 4. Using: tag_no
Select: tag_name, ec_no
From: Attribute_Use_Cl

Step 5. Using: ec_no
Select: ec_name
From: Entity_Name

View - Entity Class Mapping

Step 1. Using: N/A
Select: view_no, view_id
From: User_view

Step 2. Using: view_no
Select: ec_no
From: View_Ec_Xref

Step 3. Using: ec_no
Select: ec_name
From: Entity_Name

Complex Mapping:

Step 1. Using: N/A
Select: view_no, view_id
From: User_View

Step 2. Using: view_no
Select: di_id, di_no
From: Data_Item

Step 3. Using: di_no
Select: mod_id, parm_id, di_alg_use_code,
di_mod_instance
From: Di_Parm

Complex Mappings

Module Definition

Step 1. Using: N/A
Select: mod_id, lang_name, latest_rev_date,
status_ind
From: Software_Module

Module Parameter Definition

Step 1. Using: mod_id
Select: parm_id, parm_name, data_type_name
From: Module_parameter

Module Parameter Specification - (If Attribute)

Step 2. Using: parm_id
Select: tag_no, auc_alg_use_code,
auc_mod_instance
From: Auc_Parm

Step 3. Using: tag_no
Select: tag_name, ec_no
From: Attribute_Use_Cl

Step 4. Using: ec_no
Select: ec_name
From: Entity_Name

Module Parameter Specification - (If Constant)

Step 2. Using: parm_id
Select: constant_value, const_alg_use_code,
const_mod_instance
From: Const_Parm

Step 3. Using: parm_id
Select: rt_no, rt_alg_use_code,
rt_mod_instance
From: rt_parm

Module Parameter Specification - (If Data Field)

Step 2. Using: parm_id
Select: df_no, df_alg_use_code,
df_mod_instance
From: Df_Parm

Step 3. Using: df_no
Select: df_id, rt_id, db_id
From: Data_Field

Step 4. Using: db_id
Select: db_name
From: Data_Base

Module Parameter Specification - (If Data Item)

Step 2. Using: parm_id
Select: di_no, di_alg_use_code,
di_mod_instance
From: Di_Parm

Step 3. Using: di_no
Select: di_id, view_no
From: Data_Item

Step 4. Using: view_no
Select: view_id
From: User_View

Conceptual - Internal Schema Mappings

Attribute Class, Record Set Mapping

Step 1. Using: N/A
Select: db_id, tag_no, pref_no, map_type
From: AuC_Is_Mapping

- Step 2. Using: db_id, tag_no
Select: db_name, set_id, auc_value, tag_name,
ec_no
From: Data_Base, Auc_St_Mapping,
Attribute_Use_Cl
- Step 3. Using: ec_no
Select: ec_name, model_no
From: Entity_Name
- Step 4. Using: model_no
Select: model_name
From: Model_Class

Attribute Class, Data Field Mapping

- Step 1. Using: N/A
Select: db_id, rt_id, tag_no, pref_no, map_type
From: Auc_Is_Mapping
- Step 2. Using: db_id, tag_no
Select: db_name, ec_no, tag_name,
From: Data_Base, Attribute_Use_Cl
- Step 3. Using: ec_no, rt_id
Select: df_id, ec_name, model_no
From: Entity_Name, Project_Data_Field
- Step 4. Using: model_no
Select: model_name
From: Model_Class

Attribute Class, Complex Mapping

- Step 1. Using: N/A
Select: db_id, rt_id, tag_no, pref_no, map_type
From: Auc_Is_Mapping
- Step 2. Using: db_id, tag_no
Select: db_name, ec_no, tag_name, mod_id,
parm_id
From: Data_Base, Auc_Parm, Attribute_Use_Cl
- Step 3. Using: ec_no, mod_id, parm_id
Select: ec_name, model_no, parm_name
From: Entity_Name
- Step 4. Using: model_no
Select: model_name
From: Model_Class

Relation Class, Record Set Mapping

- Step 1. Using: N/A
Select: rc_name, ind_ec_no, dep_ec_no, rc_no
From: LINK_RELATION

Step 2. Using: ind_ec_no, dep_ec_no, rc_no
Select: ec_name(dep), ec_name(dep), model_no,
db_id, set_id, rt_id
From: Entity_Class, Rc_Based_Rec_Set

Step 3. Using: db_id, model_no
Select: db_name, model_name
From: Data_Base, Model_Class

Entity_Class, Partition Mapping

Step 1. Using: N/A
Select: ec_no, model_no
From: Entity_Class

Step 2. Using: ec_no, model_no
Select: ec_name, model_name, rt_no, hp_no
From: Entity_Name, Model_Class,
Horizontal_Partition

Step 3. Using: rt_no
Select: db_id, rt_id
From: Record_Type

Step 4. Using: db_id
Select: db_name
From: Data_Base

Entity Class, Record Type, Union Discriminator

Step 1. Using: N/A
Select: ec_no, model_no
From: Entity_Class

Step 2. Using: ec_no, model_no
Select: ec_name, model_name, db_id, rt_id,
df_id, union_value, comparison_value
From: Entity_Name, Model_Class, Ecrtud

Step 3. Using: db_id
Select: db_name
From: Data_Base

3.2.1.3 Outputs

The output from the CDM Extract will be a data file with records of the following general format.

CDM id	schema	category	level	base data	attr. data
--------	--------	----------	-------	-----------	------------

Figure 3-5 Record Format

Each field is defined as follows:

CDM Version Id - the 10 character name provided by the user to identify the CDM definition.

- schema - a 4 character designation used to identify the type of object contained in the data record.
- "0003" - Internal Schema
 - "0001" - External Schema
 - "0002" - External Schema
 - "0006" - Conceptual-Internal Schema Mapping
 - "0004" - Conceptual-External Schema Mapping
 - "0005" - Complex Mapping
- category - a 4 byte numeric field identifying a class within the schema. An example would be the category Entity Class within the Conceptual Schema. The schema identifier above would be "CS" and the category identifier would be a number representing Entity.
- level - a 4 byte numeric field identifying the level within the schema category. An example would be the level "owned attributes" belonging to the category Entity Class within the schema Conceptual Schema. The level identifier would be a number identifying "owned attributes".
- base data - a field or combination of fields which uniquely identifies a schema.category.level occurrence. The base data is the fundamental unit of comparison for determining whether a schema.category.level occurrence exists in one CDM and not in another. To check for the occurrence of an "owned attribute" for an Entity Class definition, the base data would be composed of "Model Name, Entity Class Name, Attribute Name". If a record with the corresponding base data value does not exist in the other CDM extract data file, then it can be said that this occurrence is missing from the second CDM.
- attribute data - an attribute or combination of attributes which belong to the schema.category.level named by the base data. The attribute data field is the unit of comparison which determines whether a schema.category.level occurrence in one CDM is different than the same occurrence in another CDM.

The base and attribute data areas are of variable length in general. However, within a particular schema.category.level they are fixed length. Appendix A contains the detailed definition for each possible data record.

3.2.2 CDM Compare Phase

This part of the CDM Compare Utility performs a comparison on the two files created from the extract phase. The output from the compare populates the results table. This table contains all the results of the compare.

3.2.2.1 Inputs

When the compare phase is selected from the IISS function screen, the user will be presented with the form shown in figure 3-6. The user must enter the extract file names that were supplied during the extract phase of the utility.

```
+-----+
|                                     |
|             INTEGRATED INFORMATION SUPPORT SYSTEM             |
|                                     |
|             CDM COMPARE UTILITY                                 |
|                                     |
|             ** COMPARE PHASE **                                |
|                                     |
|             Enter the names of the extract files              |
|             to be compared and press <enter>.                 |
|                                     |
|             Extract File 1: _____                        |
|                                     |
|             Extract File 2: _____                        |
|                                     |
|             Msg: 0                                           application                               |
+-----+
```

Figure 3-6 Compare Phase Input Screen

3.2.2.2 Processing

The extract files will be sorted, using a system sort, in ascending sequence for the length of the entire data record. The differences between the two versions of the CDM will be identified by comparing the base data and attribute data within a given schema.category.level. Definitions missing or different in either version of the CDM will be reported.

3.2.2.3 Outputs

The output from the compare phase will be data which populates the results table Compare_Results. The data which will go into the results table will be: CDM version, schema id, category, level, base data and description of the difference for this base data occurrence.

3.2.3 CDM Compare Report Phase

The report phase retrieves the comparison results from the Compare_Results table and outputs the results through the Report Writer to any device supported by the Report Writer. The Compare_Results table must be created prior to using the CDM Compare utility the first time. The table should remain available.

3.2.3.1 Input

When the report phase is selected from the IISS Function Screen, the user will be presented with the form shown in Figure 3-7. The user will enter the output desired output device name. The report phase also uses as input the results table populated during the compare phase.

```
+-----+
|                                     |
|             INTEGRATED INFORMATION SUPPORT SYSTEM             |
|                                     |
|             CDM COMPARE UTILITY                                 |
|                                     |
|             **  REPORT PHASE  **                                |
|                                     |
|                                     |
|             Enter the output medium and press <enter>.         |
|                                     |
|             Output Medium: _____                         |
|                                     |
|                                     |
|             Msg: 0                                           application                               |
|                                     |
+-----+
```

Figure 3-7 Report Phase Input Screen

3.2.3.2 Processing

The Compare_Results table will be read, using NDML, selecting schema, category, level, base data and description for display.

3.2.3.3 Output

The results report will be presented to the user as shown in Figure 3-8.

```
+-----+
|                                     |
|             CDM COMPARE REPORT                                 |
|                                     |
|             Conceptual Schema                                 |
|                                     |
|             Entity Class - Owned Attributes                   |
|                                     |
|             Item: model_name.ec_name.ac_name1                |
|             Reason: not defined for CDM 2                     |
|                                     |
|             Item : model_name.ec_name.ac_name2                |
|             Reason: different for CDM 2                       |
|                                     |
|                                     |
|             Msg: 0                                           application                               |
|                                     |
+-----+
```

Figure 3-8 Report Phase Output Display

3.3 Performance Requirements

3.3.1 Programming Methods

The programs consist of the functions in section 3.2 which compare an object in one version of a CDM with another. The programs also consist of a report section which formats and presents the results of the CDM Compare.

3.3.2 Modification Requirements

This system makes use of the structure of the CDM. When the CDM is modified, this development specification should be reviewed to determine if the CDM Compare requires modification.

3.4 Database Requirements

The CDM Compare makes use of the CDM database. Queries are made against the following tables: Domain_Class, Dbms, Host, User_Def_Data_Type, User_View, Data_Item, Attribute_Class, Attribute_Name, Entity_Class, Entity_Name, Relation_Class, LINK_RELATION, Data_Base, Record_Type, Data_Field, Record_Set, Ecrtud, Horizontal_Part, Project_Data_Field, Auc_St_Mapping, Rc_Based_Rec_Set, Project_Data_Item, and Model_Class, Key_Class, Attribute_Use_Cl, Owned_Attribute, Inherited_Attribute_Use, Rc_Keyword, Keyword, Domain_Range, Domain_Value, User_View, Data_Item, Schema_Names, Data_Base_Assignment, Elementary_Data_Field, Index_Df, Data_Field_Redefinition, Component_Data_Field, Repeating_Df_Occ_Counter, Data_Field_Filler, Set_Type_Member, Df_Set_Linkage, Psb_Pcb, Psb, Di_Parm, Software_Module, Module_Parameter, Auc_Parm, Const_Parm, DF_Parm, Auc_Is_Mapping.

SECTION 4

QUALITY ASSURANCE PROVISIONS

4.1 Introduction and Definitions

"Testing" is a systematic process that may be preplanned and explicitly stated. Test techniques and procedures may be defined in advance and a sequence of test steps may be specified.

"Debugging" is the process of isolation and correction of the cause of an error.

"Antibugging" is defined as the philosophy of writing programs in such a way as to make bugs less likely to occur and when they do occur, to make them more noticeable to the programmer and the user. In other words, as much error checking as is practical and possible in each routine should be performed.

4.2 Computer Programming Test and Evaluation

The quality assurance provisions for test consists of the normal testing techniques that are accomplished during the construction process. They consist of design and code walk-throughs, unit testing, and integration testing. These tests are performed by the design team. Structured design, design walk-through and the incorporation of "antibugging" facilitate this testing by exposing and addressing problem areas before they become coded "bugs".

The test will consist of making a copy of a CDM (model only) and adding, modifying or deleting one or more of each of the objects listed in section 3.2. Next the CDM compare will be run. The report should list exactly those objects added, modified or deleted.

SECTION 5

PREPARATION FOR DELIVERY

The implementation site for the constructed software is the Integrated Information Support System (IISS) Test Bed site. Each CPCI release is clearly identified and includes instructions on procedures to be followed for installation of the release. Integration with the other IISS CPCI's will be done on the IISS test bed on a scheduled basis. Delivery of this software will be on a media which is compatible with the IISS Test Bed.

APPENDIX A

EXTRACT DATA FILE RECORD DEFINITIONS

This appendix contains the individual record definitions for the records created during the extract phase of the CDM Compare utility.

A.1 CONCEPTUAL SCHEMA (schema = "0001")

A.1.1 Category 1 - Model Definition

A.1.1.1 Level 1 Description: Model Definition
Base data: model_name
Attribute data: none

A.1.2 Category 2 - Entity Class

A.1.2.1 Level 1 Description: Key Definition
Base data: model_name, ec_name, kc_name
Attribute data: tag_name

A.1.2.2 Level 2 Description: Owned Attributes
Base data: model_name, ec_name, ac_name
Attribute data: none

A.1.2.3 Level 3 Description: Inherited Attributes
Base data: model_name, ec_name, tag_name
Attribute data: ac_name

A.1.2.4 Level 4 Description: Entity as Independent Entity in
LINK_RELATION Definition
Base data: model_name, ec_name, rc_name
Attribute data: none

A.1.2.5 Level 5 Description: Entity as Dependent Entity in
LINK_RELATION Definition
Base data: model_name, ec_name, rc_name
Attribute data: none

A.1.3 Category 3 - Relation Class

A.1.3.1 Level 1 Description: Relation Definition
Base data: model_name
Attribute data: ec_name(ind), ec_name(dep)
no_ind_ent, min_no_dep_ent,
max_no_dep_ent

A.1.3.2 Level 2 Description: relation keywords
Base data: model_name, rc_name, keyword
Attribute data: none

A.1.3.3 Level 3 Description: migrating keys
Base data: model_name, rc_name,
kc_name, tag_name
Attribute data: none

A.1.4 Category 4 - Attribute Class

A.1.4.1 Level 1 Description: Attribute Definition
Base data: model_name, ac_name
Attribute data: ac_name_type, domain_name,
keyword

A.1.4.2 Level 2 Description: Attribute keywords
Base data: model_name, ac_name, keyword
Attribute data: none

A.1.5 Category 5 - Domains

A.1.5.1 Level 1 Description: Data Type
Base data: domain_name
Attribute data: data_type_name, max_size,
no_of_decimals, data_type_ind

A.1.5.2 Level 2 Description: Domain Range
Base data: domain_name
Attribute data: begin_value, end_value

A.1.5.3 Level 3 Description: Domain Value
Base data: domain_name
Attribute data: specific_value

A.2 EXTERNAL SCHEMA (schema = "0002")

A.2.1 Category 1 - User Views

A.2.1.1 Level 1 Description: User View Definition
Base data: view_id
Attribute data: di_id, data_type_name

A.3 INTERNAL SCHEMA (schema = "0003")

A.3.1 Category 1 - Data Base Definition

A.3.1.1 Level 1 Description: Data Base Environment
Base data: db_name
Attribute data: dbms_name, host_id, null_value
schema_name, subschema_name,
db_location

A.3.1.2 Level 2 Description: Area Assignments
Base data: db_name, area_id
Attribute data: rt_id

A.3.1.3 Level 3 Description: Record Type Area Assignments
Base data: db_name, rt_id
Attribute data: area_id

- A.3.1.4 Level 4 Description: Record Type Data Field Definitions
Base data: db_name, rt_id, df_id
Attribute data: rec_seq_no, rec_key_code,
occurs, dbms_access,
index_indicator,
data_type_name,
redef_df_no(name),
component_of_df(name),
occ_depend_df_no(name),
index_by_df_name, index_of_df_name, filler_size
- A.3.1.5 Level 5 Description: Record type data field redefined by
Base data: db_name, rt_id, df_id
Attribute data: redef_df_name
- A.3.1.6 Level 6 Description: Record type data field composed of
Base data: db_name, rt_id, df_id
Attribute data: composed_of_name
- A.3.1.7 Level 7 Description: Record Set Definition
Base data: db_name, set_id, rt_id_of_owner,
Attribute data: total_num_mem, req_mem_ind,
rt_id_of_member,
df_id, linkage_type
- A.3.1.8 Level 8 Description: Data Base PSB Definition
Base data: db_name, psb_name
Attribute data: pcb_seq_no, key_feedback_len,
host_id
- A.4 CONCEPTUAL - EXTERNAL SCHEMA MAPPINGS (schema = "0004")
- A.4.1 Category 1 - External To Conceptual Schema Mappings
- A.4.1.1 Level 1 Description: Project Data Item Mapping
Base data: view_id, di_id
Attribute data: ec_name, tag_name,
prim_secondary
- A.4.1.2 Level 2 Description: View - Relation Class Mapping
Base data: view_id, rc_name
Attribute data: ec_name(ind), ec_name(dep)
- A.4.1.3 Level 3 Description: Complex Mapping
Base data: view_id, di_id
Attribute data: mod_id, parm_id,
di_alg_use_code, di_mod_instance
- A.5 COMPLEX MAPPING (schema = "0005")
- A.5.1 Category 1 - Definitions
- A.5.1.1 Level 1 Description: Module Definition
Base data: mod_id
Attribute data: language, latest_rev_date,
status_ind

- A.5.1.2 Level 2 Description: Parameter Definition
Base data: mod_id, parm_name
Attribute data: parm_id, data_type_name
- A.5.1.3 Level 3 Description: Parameter Specification
Base data: mod_id, parm_name
Attribute data:
if attribute:
 ec_name, tag_name,
 auc_alg_use_code,
 auc_mod_instance
if constant:
 constant_value,
 const_alg_use_code,
 const_mod_instance
if record type:
 db_name, rt_id,
 rt_alg_use_code,
 rt_mod_instance
if data field:
 db_name, rt_id, df_id,
 df_alg_use_code,
 df_mod_instance
if data item:
 view_id, di_id,
 di_alg_use_code,
 di_mod_instance
- A.6 CONCEPTUAL - INTERNAL SCHEMA MAPPINGS (schema = "0006")
- A.6.1 Category 1 - Attribute Class Mappings
- A.6.1.1 Level 1 Description: Record Set Mapping
Base data: model_name, ec_name, tag_name
Attribute data: pref_no, db_name, set_id,
auc_value
- A.6.1.2 Level 2 Description: Data Field Mapping
Base data: model_name, ec_name, tag_name
Attribute data: pref_no, db_name, rt_id,
df_id
- A.6.1.3 Level 3 Description: Complex Mapping
Base data: model_name, ec_name, tag_name
Attribute data: pref_no, mod_id, parm_name,
db_name, rt_id
- A.6.2.1 Category 2 - Relation Class Mapping
- A.6.2.1 Level 1 Description: Relation - Record Set Mapping
Base data: model_name, ec_name, rc_name
Attribute data: ec_name(dep), db_name, set_id,
rt_id_of_member
- A.6.3.1 Category 3 - Entity Class Mappings

- A.6.3.1 Level 1 Description: Partition Mapping
Base data: model_name, ec_name
Attribute data: hp_no, db_name, rt_id
- A.6.3.2 Level 2 Description: Entity Class - Record Type -
Union Discriminator
Base data: model_name, ec_name
Attribute data: db_name, rt_id, df_id,
comparison_op, union_value